PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Luc Van Brabant	Technology Center: 2100
Serial No.: 10/748,008 Confirm 6494	Group Art Unit: 2139
Filed: 12/30/2003	Examiner: Wang, Harris C.
For: ON-ACCESS AND ON-DEMAND DISTRIBUTED VIRUS SCANNING	Atty. Dkt. No.: 10830.0103.NP

APPEAL BRIEF TO THE BOARD OF PATENT APPEALS AND INTERFERENCES

Commissioner for Patents
PO Box 1450
Alexandria, Virginia 22313-1450

Sir:

Please deduct any deficiency in the required fee for this Appeal Brief from EMC Corporation Deposit Account No. 05-0889.

I. REAL PARTY IN INTEREST

The real party in interest is EMC Corporation, by virtue of an assignment recorded at Reel 014859 Frame 0273.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

III. STATUS OF THE CLAIMS

Claims 1-28 have been presented for examination.

Claims 1-28 have been finally rejected, and are being appealed.

IV. STATUS OF AMENDMENTS

No amendment was filed after the final Official Action of Sep. 26, 2007.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The applicant's invention relates generally to "on-access" virus scanning and "on-demand" virus scanning. "On-access" virus scanning occurs when a specified trigger occurs, such as when a user accesses a file marked "unchecked." (Appellant's specification, page 4, lines 7-8.) "On-demand" virus scanning is typically scheduled when a new virus is discovered, when new unchecked files are migrated to a file server, or prior to archiving or backing-up unchecked files. (Appellant's specification, page 4, lines 8-10.)

The appellant's invention of independent claim 1 provides a method of operating a plurality of virus checkers (32, 33, and 34 in FIG. 1 and FIG. 3) for on-demand anti-virus scanning concurrent with on-access anti-virus scanning. (Appellant's specification, page 4, line 21-23; page 9 lines 3-7). The method of claim 1 includes combining on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue (63 in FIG. 3), and distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers. (Appellant's specification, page 4, line 23 to page 5, line 2; page 13 lines 13-15; page 13 lines 1-7). For example, the on-demand anti-virus scan requests and the on-access anti-virus scan requests are distributed from the virus scan request queue to the virus checkers by AV threads in a pool (64 in FIG. 3), and each AV thread is programmed as shown in steps 71 to 83 of FIGS. 4 and 5 to send the request from the head of the virus scan request queue (step 76 in FIG. 4) to a particular one of the virus checkers (step 83 in FIG. 5) assigned by a distribution list (62 in FIG. 3), as described in appellant's specification on page 13 lines 1-4 and page 13 line 23 to page 14 line 21.

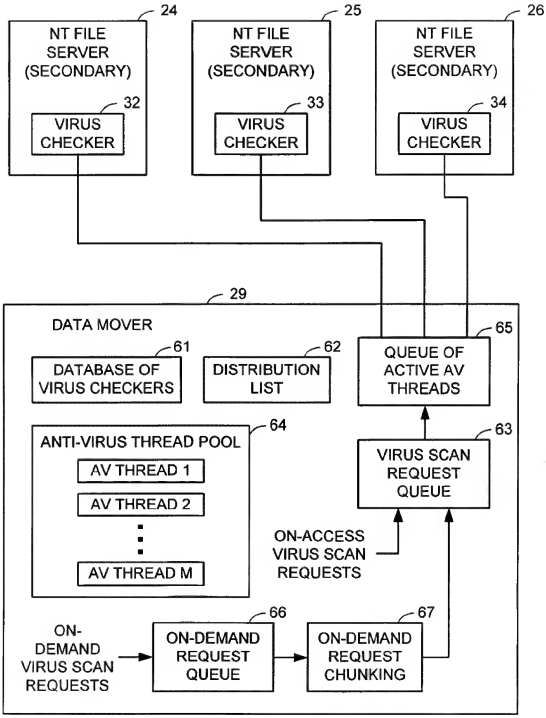
24
NT FILE SERVER (SECONDARY)
32
VIRUS CHECKER
25
NT FILE SERVER (SECONDARY)
33
VIRUS CHECKER
26
NT FILE SERVER (SECONDARY)
34
VIRUS CHECKER
29
DATA MOVER
61
DATABASE OF VIRUS CHECKERS
62
DISTRIBUTION LIST
65
QUEUE OF ACTIVE AV THREADS
64
ANTI-VIRUS THREAD POOL
AV THREAD 1
AV THREAD 2
AV THREAD M
63
VIRUS SCAN REQUEST QUEUE
ON-ACCESS VIRUS SCAN REQUESTS
66
ON-DEMAND REQUEST QUEUE
67
ON-DEMAND REQUEST CHUNKING
ON-DEMAND VIRUS SCAN REQUESTS

FIG. 3

The appellant's invention of independent claim 8 provides a method of operating a plurality of virus checkers (32, 33, and 34 in FIG. 1 and FIG. 3). (Appellant's specification, page 5, lines 3-4; page 9 lines 3-7.) The method includes distributing on-demand anti-virus scan requests and on-access anti-virus scan requests to the virus checkers so that the virus checkers perform on-demand anti-virus scanning concurrent with on-access anti-virus scanning. (Appellant's specification, page 5, lines 4-7.) The method further includes grouping the on-demand anti-virus scan requests into chunks of multiple ones of the on-demand anti-virus scan requests (67 in FIG. 3; steps 108-109 in FIG. 7), and for each chunk, distributing the multiple ones of the on-demand anti-virus scan requests over the virus checkers. (Appellant's specification, page 5, lines 7-10; page 13 lines 13-17 and 20-22; page 16, lines 18-22.)

The appellant's invention of independent claim 12 provides a method of operating a plurality of virus checkers (32, 33, and 34 in FIG. 1 and FIG. 3) for on-demand anti-virus scanning concurrent with on-access anti-virus scanning. (Appellant's specification, page 5, lines 11-13; page 9 lines 3-7.) The method includes combining on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue (63 in FIG. 3), and a pool of threads (64 in FIG. 3) distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers. (Appellant's specification, page 5, lines 13-16.) Each anti-virus scan request on the virus scan request queue is serviced by a respective one of the threads in the pool of threads. (Appellant's specification, page 5, lines 17-18.) The method further includes grouping the on-demand anti-virus scan requests into chunks of multiple ones of the on-demand anti-virus scan requests (67 in FIG. 3;

steps 108-109 in FIG. 7), and for each chunk, checking whether the number of anti-virus scan requests on the virus checking queue is less than a threshold (TH1 in step 124 of FIG 8), and upon finding that the number of anti-virus scan requests on the virus checking queue is less than the threshold, placing the chunk on the virus scan request queue (step 126 in FIG. 8). (Appellant's specification, page 5, lines 18-23; page 13 lines 13-17; page 17 line 21 to page 18 line 3.)

The appellant's invention of independent claim 16 provides a virus checking system including a plurality of virus checkers (32, 33, and 34 in FIG. 1 and FIG. 3) for on-demand anti-virus scanning concurrent with on-access anti-virus scanning, a virus scan request queue (63 in FIG. 3), and at least one processor (29 in FIG. 3) coupled to the virus checkers and the virus scan request queue for sending virus scan requests from the virus scan request queue to the virus checkers. (Appellant's specification, page 6, lines 2-6; page 9 lines 3-7; page 8 line 22 to page 9 line 1.) The at least one processor is programmed for placing on-demand anti-virus scan requests and on-access anti-virus scan requests onto the virus scan request queue, and for distributing the on-demand anti-virus scan requests and the on-access virus scan requests from the virus scan request queue to the virus checkers. (Appellant's specification, page 6, lines 6-10.)

The appellant's invention of independent claim 24 provides a virus checking system including a plurality of virus checkers (32, 33, and 34 in FIG. 1 and FIG. 3) for on-demand anti-virus scanning concurrent with on-access anti-virus scanning, and a file server (27 in FIG. 1) coupled to the virus checkers for sending virus checking requests from the file server to the virus checkers. (Appellant's specification, page 6, lines 11-15; page 8 lines 8-11 and 17-18; page 9

lines 3-7.) The file server includes a virus scan request queue (63 in FIG. 3). (Appellant's specification, page 6, line 15; page 12 lines 13-15; page 13, lines 2-7.) The file server is programmed for placing on-demand anti-virus scan requests and on-access anti-virus scan requests onto the virus scan request queue, and for executing multiple threads (64 in FIG. 3) for distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers (steps 71 to 83 of FIGS. 4 and 5). (Appellant's specification, page 6, lines 15-19; page 13 lines 2-17; page 13 line 23 to page 14 line 21.) Each anti-virus scan request on the virus scan request queue is serviced by a respective one of the threads in the pool of threads (steps 71 to 83 of FIGS. 4 and 5). (Appellant's specification, page 6, lines 19-21; page 13 line 23 to page 14 line 21.) The file server is further programmed for grouping the on-demand anti-virus scan requests into chunks (67 in FIG. 3; steps 108-109 in FIG. 7) of multiple ones of the on-demand anti-virus scan requests, and for consecutively placing the chunks onto the virus scan request queue. (Appellant's specification, page 6, lines 21-23; page 13 lines 13-17 and 20-22; page 16, lines 18-22.)

None of appellant's claims contain any "means plus function" or "step plus function" as permitted by 35 U.S.C. 112, sixth paragraph.

Appellant's dependent claims 5 and 21 further define giving the on-access anti-virus scan requests priority over the on-demand anti-virus scan request by inhibiting the placement of on-demand anti-virus scan requests onto the virus scan request queue (63 in FIG. 3) when the number of anti-virus scan requests on the virus scan request queue reaches a threshold (TH1 in step 124 of FIG 8), and not inhibiting the placement of on-access anti-virus scan requests onto

the virus scan request queue when the number of anti-virus scan requests on the virus scan request queue reaches the threshold. (Steps 124 and 126 in FIG. 8; appellant's specification, page 12, 13-20; page 13 lines 8-17; page 17 line 21 to page 18 line 3.) In a similar fashion, appellant's dependent claim 27 further defines checking for each chunk whether the number of anti-virus scan requests on the virus checking queue is less than a threshold (TH1 in step 124 of FIG 8), and upon finding that the number of anti-virus scan requests on the virus checking queue is less than the threshold, placing said each chunk on the virus scan request queue. (Steps 124 and 126 in FIG. 8; appellant's specification, page 12, 13-20; page 13 lines 8-17; page 17 line 21 to page 18 line 3.)

Appellant's dependent claims 6 and 22 further define grouping the on-demand anti-virus scan requests into chunks (67 in FIG. 3), each of the chunks including multiple ones of the on-demand anti-virus scan requests (steps 108-109 in FIG. 7), and placing the chunks onto the virus scan request queue (step 110 in FIG. 7; step 126 in FIG. 8). (Appellant's specification, page 12 lines 20-23; page 13 lines 13-17 and 20-22; page 16, lines 18-22; page 17, lines 12-14.)

Appellant's dependent claims 7, 11, 15, 23, and 28 further define inhibiting the placement of at least one of the chunks onto the virus scan request queue until completion of anti-virus scanning for the anti-virus scan requests in a prior one of the chunks (step 110 in FIG. 7; steps 121, 122, and 123 in FIG. 8). (Appellant's specification, page 12 lines 20-23; page 16, lines 18-22; page 17, line 12 to page 18, line 3.)

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Whether claims 1-28 are unpatentable under 35 U.S.C. 103 over Edwards U.S. Patent 7,188,367 in view of Novell's article: AntiVirus Solutions for Network, January 1, 1999, pgs. 1-4.

VII. ARGUMENT

Claims 1-28 are patentable under 35 U.S.C. 103 over Edwards U.S. Patent 7,188,367 in view of Novell's article: AntiVirus Solutions for Network, January 1, 1999, pgs. 1-4.

The policy of the Patent and Trademark Office has been to follow in each and every case the standard of patentability enunciated by the Supreme Court in Graham v. John Deere Co., 148 U.S.P.Q. 459 (1966). See M.P.E.P. § 2141; KSR International Co. v. Teleflex Inc., 550 U.S. __, 82 USPQ2d 1385 (2007). As stated by the Supreme Court:

> Under § 103, the scope and content of the prior art are to be determined; differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background, the obviousness or nonobviousness of the subject matter is determined. Such secondary considerations as commercial success, long felt but unsolved needs, failure of others, etc., might be utilized to give light to the circumstances surrounding the origin of the subject matter sought to be patented. As indicia of obviousness or nonobviousness, these inquiries may have relevancy.

148 U.S.P.Q. at 467.

Scope and Content of the Prior Art

Edwards discloses a virus scanner in which a pool of pre-processor threads and a queue are interposed between the event filter and the pool of scanner threads. The pre-processor threads perform operations that can be completed quickly to determine whether an object of a scan request needs to be scanned. The pre-processor threads gather characteristics about the scan requests and place them in the queue in a priority order based on those characteristics. The scanner threads select a scan request from the queue based on the priority order. Alternatively, the scan request is selected based on the scan request's characteristics as compared to the characteristics of the scan requests whose objects are currently being scanned by other scanner threads in the pool. (Edwards, Abstract.) Virus scanners may be invoked on-demand by a computer user to scan a selected file. More typically, virus scanners install themselves as part of an operating system, and then scan files, according to user preferences, as the files are created and accessed. This type of virus scanner is referred to as an on-access virus scanner. (Edwards, col. 1, lines 42-47.)

Novell says: "Command AntiVirus with F-PROT Professional 4.52 for NetWare from Command Software Systems Inc. is an enterprise-wide antivirus solution. Command AntiVirus with F-PROT Professional 4.52 for NetWare offers a heuristic analysis, which includes both an on-access scanner and an on-demand scanner to search for polymorphic viruses. In addition, you can configure multiple concurrent scans."

Differences between the prior art and the claims at issue

Claims 1 and 16

Appellant's claim 1 defines a method of operating a plurality of virus checkers for on-demand anti-virus scanning concurrent with on-access anti-virus scanning. The method comprises combining on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue; and distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers.

Regarding the differences between the cited art and the subject matter of appellant's claim 1, page 8 of the final Official Action says that although Edwards does teach both on-demand and on-access virus scan requests, Edwards does not explicitly teach wherein the scan request queue has on-demand anti-virus scanning concurrent with on-access anti-virus scanning. Edwards, Col. 1, lines 43-48, say: "Virus scanners may be invoked on-demand by a computer user to scan a selected file. More typically, virus scanners install themselves as part of an operating system and then scan files, according to user preferences, as the files are created and accessed. This type of virus scanner is referred to as an on-access virus scanner."

Appellant respectfully submits that the system shown in FIG. 1 of Edwards and labeled "Prior Art" is said to be an on-access virus scanner. Edwards, col. 2, lines 12-24, say:

> A prior art on-access virus scanner 100 is organized into two parts, as illustrated in FIG. 1. One part is the event filter 110, which is the software that

> intercepts the events of interest to the virus scanner. Events of interest include a file being opened or an e-mail arriving in a mailbox. Another part is a scanner thread 120, which is the software that receives scan requests from the filter. The scanner thread determines whether the object of the intercepted event (i.e. the file, e-mail, or e-mail attachment) needs scanning and, if so, scans the object. Multiple scanner threads are typically provided in pools 130 that are capable of executing concurrently so that multiple objects may be scanned simultaneously.

As further disclosed in Edwards column 2, lines 25-37, Edwards is directed to a problem that arises in the on-access virus scanner:

> Unfortunately, virus developers have recently begun to manufacture "malicious" files which take "a long time" to scan, including archives and documents containing embedded objects. The malicious files are designed to overwhelm on-access virus scanners by tying up all of the available scanner threads in the pool, thereby causing all other events intercepted by the filter to be queued until a scanner thread becomes free. This causes the virus scanner to "crash" by blocking further processing of data and leaves a system undefended against subsequent attacks. If e-mail or file processing is routed through a virus scanner and the scanner has crashed, then a "denial of service" for e-mail or file activity occurs until the scanner is restarted.

Edward's embodiment disclosed in FIGS. 2-6 also appears to be directed to an on-access virus scanner because the disclosed embodiment includes an event filter (Edwards, FIG. 2, box 110). Edwards col. 4, lines 50-53 say: "A scan request is generated whenever an event occurs that causes the host system to access an object, such as opening a file, reading an e-mail, or

opening an e-mail attachment." Edwards col. 4 lines 64-67 say: "As illustrated, a pre-processor pool 230 of four pre-processor threads 210 and a priority queue 220 are interposed between the event filter 110 and the scanner thread pool 130 of three scanner threads 120." Edwards col. 10, lines 3-7 say: "For example, while the foregoing description focused on on-access virus scanners, it will be recognized that the above techniques and analyses can be applied to scanning data in other contexts such as on-demand virus scanners having comparable limitations."

Novell also teaches that on-demand scanners are different from on-access scanners. Although Novell says you can configure multiple concurrent scans, there is no suggestion that the on-access scanner would ever concurrently scan for an on-demand anti-virus scan request, nor is there any suggestion that the on-demand scanner would ever concurrently scan for an on-access anti-virus scan request.

Page 9 of the final Official Action says it would have been obvious to one of ordinary skill in the art to combine the virus scan queue of Edwards to include concurrent on-access and on-demand scanners. Appellant respectfully disagrees. Edwards teaches that a particular problem with the Edwards FIG. 1 prior art on-access anti-virus scanner can be solved by adding a priority queue 220 to produce the on-access antivirus scanner shown in Edwards FIG. 2. Although it would be possible for an on-demand anti-virus scanner to have a queue, it is not understood how the virus scan queue of Edwards would or should be modified to include an on-access anti-virus scanner and an on-demand anti-virus scanner. More importantly, appellant's claim 1 is not calling for a virus scan queue including concurrent on-access and on-demand scanners. Instead, appellant's claim 1 calls for combining on-demand anti-virus scan requests

and on-access anti-virus scan requests in a virus scan request queue, and distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers (for on-demand anti-virus scanning concurrent with on-access anti-virus scanning).

Page 9 of the final Official Action says: "The motivation to combine is that it is well known in the virus scanning art to have both on-demand and on-access scanners running concurrently. Novell provides an example of such a system." Although it should be possible to run the on-demand scanner of Novell concurrently with the on-access scanner of Novell in a multi-processing computer system, this would not have motivated one of ordinary skill to combine on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue, and distribute the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the Novell on-demand checker and the Novell on-access checker running concurrently. From the teaching in Novell, one would simply send on-demand anti-virus scan requests to the Novell on-demand scanner, and the Novell on-access scanner would scan in response to any on-access events. If the Novell on-access scanner would have a problem as described in Edwards, then the Novell on-access scanner could be modified as described in Edwards, but that would not result in combining on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue, as called for in appellant's claim 1.

"[R]ejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." In re Kahn, 441 F. 3d 977, 988 (Fed. Cir. 2006). A fact finder should be aware of the distortion caused by hindsight bias and must be cautious of arguments reliant upon ex post reasoning. See KSR v. Teleflex, 550 U.S. __ (2007), citing Graham, 383 U. S. at 36 (warning against a "temptation to read into the prior art the teachings of the invention in issue" and instructing courts to "guard against slipping into the use of hindsight.").

The problem that the inventor is trying to solve must be considered in determining whether or not the invention would have been obvious. The invention as a whole embraces the structure, properties and problems it solves. In re Wright, 848 F.2d 1216, 1219, 6 U.S.P.Q.2d 1959, 1961 (Fed. Cir. 1988). In contrast to Edwards and Novel, the appellant's specification teaches that it is desirable to use the same virus checkers for on-demand and on-access virus checking. This leads to several problems because the priority and workload for the on-demand scan requests are so much different from the priority and workload for the on-access scan requests. The different priority and workload would suggest that the on-demand scan requests should be treated differently from the on-access scan requests. Nevertheless, the applicant's specification teaches that it is desirable to combine the on-demand requests and the on-access requests in a queue under appropriate conditions. This is explained in appellant's specification on page 11 line 20 to page 12 line 23 as follows:

[00030] The scanning task shown in FIG. 2 is generally referred to as an "on-access" virus scan. An "on-access" virus scan is processed in real-time when scanning is triggered by user-initiated file access. Another kind of virus scan is known as an "on-demand" virus scan. An "on-demand" virus scan is scheduled at a lower priority than "on-access," and it typically involves scanning all files of virus-checkable file type in a one or more specified file systems. For example, "on-demand" virus checking is scheduled when a new virus is discovered, when new unchecked files are migrated into a file server, or prior to archiving or backing-up unchecked files. Although lower in priority from a scheduling point of view, "on-demand" virus checking has often been more burdensome on the data processing system than "on-access" virus checking. A full file system scan may generate a much more intense scanning load when a multitude of files in the file system must be scanned. Even though this scanning workload is distributed over multiple virus checkers, the volume of scans will generate a significant resource load on the operating system of the data mover. Moreover, it is desirable to fully utilize the capabilities of the virus checkers in order to complete the full file system scan as soon as possible

[00031] In order to mitigate any general performance degradation on the data mover during user file access, it is desirable to mix "on-demand" virus scan requests with "on-access" virus scan requests in a shared virus scan request queue. For example, outstanding "on-demand" virus scan requests are added to the shared queue when the number of requests in the shared queue falls below a threshold. The threshold is selected to provide a relatively continuous flow of requests to the virus checkers without significantly degrading the response time of the virus checkers for responding to the "on-access" requests. Moreover, it is desirable to add outstanding "on-demand" virus scan requests to the shared queue in manageable "chunks", and to wait until the virus scan requests in each chunk

have been serviced before sending another chunk of "on-demand" virus scan requests.

Neither Edwards nor Novell suggest these problems or the particular solution as defined in appellant's claim 1. The fact that the on-demand scan requests have priority and workloads that are so much different from the priority and workloads of the on-access scan requests, and the fact that Novell teaches that an on-demand virus checker should be used for on-demand scan requests and an on-access virus checker should be used for on-access scan requests, teach away from treating the on-demand scan requests and the on-access scan requests in a similar fashion by combining them in a virus scan request queue for distribution to virus checkers. Edwards says nothing contrary to the teaching of Novel, which is about four years prior to the filing of the appellant's patent application. Edwards in fact refers to an on-access virus scanner as being a type of virus scanner different from an on-demand virus scanner (see col. 1, lines 43-47) and having particular problems (col. 1 line 65 to col. 2 line 67) and scanning data in another context than an on-demand virus scanner (col. 10 lines 3-7). In short, the teachings of Edwards and Novell and the nature of the problem solved by the appellant show that the subject matter of appellant's claims 1 and 16 would not have been obvious.

Claims 5 and 21

Appellants' claims 5 and 21 further define that the on-access anti-virus scan requests are given priority over the on-demand anti-virus scan requests by inhibiting the placement of on-

demand anti-virus scan requests onto the virus scan request queue when the number of anti-virus scan requests on the virus scan request queue reaches a threshold, and not inhibiting the placement of on-access anti-virus scan requests onto the virus scan request queue when the number of anti-virus scan requests on the virus scan request queue reaches the threshold.

Page 10 of the final Official Action says that Edwards and Novell teach that "certain scan requests are given priority based on certain characteristics by not inhibiting a first kind of scan requests while inhibiting the placement of a second kind of scan request based on a second group of characteristics when the number of anti-virus scan requests on the virus scan request queue reaches a threshold." In particular, Edwards teaches "a pending scan request from user A may be determined to be more suitable than a pending scan request from user B if three of the four scanner threads are already scanning scan requests from user B. This prevents a single user B from monopolizing the virus scanner (Column 5, lines 66-67, Column 6, lines 1-3)."

Other passages of Edwards disclose that event filter 110 or scan prioritizer 240 places the on-access virus scan requests on the virus scan request queue, and some of the virus scan requests on the queue are given priority over other virus scan requests on the priority queue because the event filter or scan prioritizer places the virus scan requests on the queue in the order that the threads should process them, and because the scanner threads select for scanning particular virus scan requests that are on the queue based on the position of the virus scan requests on the queue or the operational characteristics of the virus scan requests. For example, Edwards col. 5, lines 39-43 say: "In one embodiment, the characteristics obtained by the pre-processor threads 210 are used by the event filter 110 to prioritize the scan requests by placing

them in the priority queue 220 in the order in which the scanner threads 120 should process them." Edwards col. 6 lines 23-29 say: "As another example, using the scan request's operational characteristics, such as a time stamp of when the scan request was triggered by the event filter 110 or when it was placed on the priority queue 220, scan requests that have been passed over too often (i.e. that have been on the priority queue 220 the longest) could eventually be given higher priority than scan requests that would otherwise come first." Edwards col. 6 lines 42-46 say: "In each of the above examples, the scanner threads 120 process scan requests either by selecting the most suitable pending request, or by selecting the next pending request that was placed on the priority queue in the optimal priority order by the scan prioritizer 240."

However, neither Edwards nor Novell gives on-access anti-virus scan requests priority over on-demand anti-virus scan requests. Nor is there any disclosure in Edwards col. 5, lines 66-67 or col. 6, lines 1-3 of inhibiting the placement of any particular kind of virus scan request onto the virus scan request queue when the number of anti-virus scan requests on the queue reaches a threshold. Inhibiting placement of a request onto the virus scan request queue is different from placing a request on the virus scan request queue in a particular order with respect to requests already on the virus request queue. Inhibiting placement of a request onto the virus scan request queue is also different from a thread selecting a particular request on the virus scan queue for distribution to one of the virus checkers. Where the prior art references fail to teach a claim limitation, there must be "concrete evidence" in the record to support an obviousness rejection. In re Zurko, 258 F.3d 1379, 1385-86, 59 U.S.P.Q.2d 1693, 1697 (Fed. Cir. 2001).

Claims 6 and 22

Appellant's claims 6 and 22 further define grouping the on-demand anti-virus scan requests into chunks, each of the chunks including multiple ones of the on-demand anti-virus scan requests, and placing the chunks onto the virus scan request queue.

Page 10 of the final Official Action says that Edwards teaches grouping individual scan requests into chunks based on characteristics, and placing the chunks onto the virus scan requests queue, and scanning one type of the grouped scans until completion before scanning a second type. In particular, Edwards column 7, lines 33-34 says: "a pending scan request from user A may be determined to be more suitable than a pending scan request from user B if three of the four scanner threads are already scanning scan request from user B. This prevents a single user B from monopolizing the virus scanner."

However, Edwards fails to disclose grouping on-demand anti-virus scan requests into chunks, and placing the chunks onto the virus scan request queue. For the reasons discussed above with respect to appellant's claim 1, placing on-demand anti-virus scan requests on Edward's priority queue of on-access anti-virus scan request is not suggested by the fact that Edwards places or selects for scanning on-access anti-virus requests on the queue based on operational characteristics of the on-access anti-virus scan requests. Nor does Edwards disclose grouping any particular kind of request into chunks, and then placing such chunks onto the virus scan request queue. Instead, as discussed above, Edwards discloses that the event filter or scan prioritizer determines whether or not a virus scan request needs a scan of an object, and if so, places each such on-access virus scan request on the priority queue, and a scanner thread selects

a virus scan request on the priority queue to process based on the position of the virus scan request on the queue and on the scan request's operational characteristics. In particular, the embodiment of Edwards in which the scan prioritizer places the scan request on the priority queue in some pre-defined fixed order other than the order in which the scan request was received, e.g. executable files first and data files second (Edwards, col. 7, lines 31-37), is different from grouping the data files into a chunk, and placing the chunk on the queue. Nor is grouping the data files into a chunk, and placing the chunk on the queue "inherent," because the method disclosed in Edwards in fact obtains executable files first and data files second in the queue without grouping the data files into a chunk, and placing the chunk on the queue. In accordance with the method disclosed in Edwards, when the scan prioritizer is placing a new request to scan an executable file in the queue and the queue already contains one or more requests for scanning data files, the scan prioritizer places the request to scan the executable file in the queue at a position before the one or more requests for scanning data files.

Claims 7 and 23

Appellant's claims 7 and 23, which are dependent respectively on claims 6 and 22, further define inhibiting the placement of at least one of the chunks onto the virus scan request queue until completion of anti-virus scanning for the anti-virus scan requests in a prior one of the chunks. This is clearly different from simply scanning one type of grouped scans before scanning another type of grouped scans. Moreover, appellant's chunks of claim 7 are of the same type, namely, chunks of on-demand virus scan requests.

Claim 8

Appellant's claim 8 calls for distributing on-demand anti-virus scan requests and on-access anti-virus scan requests to virus checkers so that the virus checkers perform on-demand anti-virus scanning concurrent with on-access anti-virus scanning. Claim 8 also calls for distributing the multiple ones of the on-demand anti-virus scan requests over the virus checkers. Thus, claim 8 calls for at least one virus checker that performs the concurrent on-access virus scanning to also perform an on-demand anti-virus scan, because the multiple ones of the on-demand anti-virus scan requests are distributed over the virus checkers. As discussed above with respect to appellant's claim 1, neither Edwards nor Novell suggests that an on-access virus checker should perform on-demand antivirus checking.

Appellant's claim 8 further calls for "grouping the on-demand anti-virus scan requests into chunks, each of the chunks including multiple ones of the on-demand anti-virus scan requests, and for each chunk, distributing the multiple ones of the on-demand anti-virus scan requests over the virus checkers." As discussed above with respect to appellant's claim 6, neither Edwards nor Novell suggests grouping of on-demand virus scan requests into chunks, and for each chunk, distributing the on-demand anti-virus scan requests over the virus checkers. In particular, Edwards giving priority to one type of on-access scans (scans of executable files) over another type of on-access scans (scans of data files) to solve a particular problem in an on-access virus checker does not suggest that on-demand virus scans should be grouped into chunks

for distribution of the on-demand virus scan requests for each chunk over virus checkers that concurrently perform on-demand and on-access virus checking.

Claim 11

Appellant's claim 11 is dependent upon claim 8, and further recites the express limitations of claim 7. Thus, claim 11 is patentable over the proposed combination of Edwards and Novel for the reasons given above with reference to claim 8 and also for the reasons given above with reference to claim 7.

Claim 12

Appellant's independent claim 12 includes limitations found in appellant's claim 1 and 6, and limitations similar to those found in appellant's claim 5. Therefore, appellant's independent claim 12 is patentable over the proposed combination of Edwards and Novel for the reasons given above with reference to claims 1, 5, and 6.

Claim 15

Appellant's claim 15 is dependent upon claim 12, and further recites the express limitations of claim 7. Thus, claim 11 is patentable over the proposed combination of Edwards and Novel for the reasons given above with reference to claim 12 and also for the reasons given above with reference to claim 7.

Claim 24

Appellants' independent claim 24 includes limitations of appellant's claims 1 and 6, and in addition, the file server is programmed for "consecutively" placing the chunks of on-demand anti-virus scan requests onto the virus scan request queue. Therefore, claim 24 is patentable over the proposed combination of Edwards and Novell for the reasons given above with reference to claims 1 and 6, and in addition, further distinguishes Edwards by consecutively placing chunks of multiple ones of the on-demand anti-virus scan requests onto the virus scan request queue instead of consecutively placing individual scan requests onto the virus scan request queue.

Claim 27

Appellant's claim 27 is dependent upon claim 24, and further recites express limitations of claim 5. Therefore, claim 27 is patentable over the proposed combination of Edwards and Novell for the reasons given above with reference to claim 24 and claim 5.

Claim 28

Appellants' claim 28 is dependent upon claim 24, and further recites the express limitations of claim 7. Therefore, claim 28 is patentable over the proposed combination of Edwards and Novell for the reasons given above with reference to claim 24 and claim 7.

In view of the above, the rejection of claims 1-28 should be reversed.

Respectfully submitted,

/ Richard C. Auchterlonie /

Richard C. Auchterlonie
Reg. No. 30,607

NOVAK DRUCE & QUIGG, LLP
1000 Louisiana, 53rd Floor
Houston, TX 77002
713-571-3400

VIII. CLAIMS APPENDIX

The claims involved in this appeal are as follows:

1. A method of operating a plurality of virus checkers for on-demand anti-virus scanning concurrent with on-access anti-virus scanning, the method comprising:

combining on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue; and

distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers.

2. The method as claimed in claim 1, wherein the on-access anti-virus scan requests are produced in response to user access to files.

3. The method as claimed in claim 1, wherein the on-demand anti-virus scan requests are produced in response to a system administrator requesting a scan of files within a specified file system.

4. The method as claimed in claim 1, wherein a pool of threads distribute the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request

queue to the virus checkers, each anti-virus scan request on the virus scan request queue being serviced by a respective one of the threads in the pool of threads.

5. The method as claimed in claim 1, wherein the on-access anti-virus scan requests are given priority over the on-demand anti-virus scan requests by inhibiting the placement of on-demand anti-virus scan requests onto the virus scan request queue when the number of anti-virus scan requests on the virus scan request queue reaches a threshold, and not inhibiting the placement of on-access anti-virus scan requests onto the virus scan request queue when the number of anti-virus scan requests on the virus scan request queue reaches the threshold.

6. The method as claimed in claim 1, which includes grouping the on-demand anti-virus scan requests into chunks, each of the chunks including multiple ones of the on-demand anti-virus scan requests, and placing the chunks onto the virus scan request queue.

7. The method as claimed in claim 6, which includes inhibiting the placement of at least one of the chunks onto the virus scan request queue until completion of anti-virus scanning for the anti-virus scan requests in a prior one of the chunks.

8. A method of operating a plurality of virus checkers, the method comprising:

distributing on-demand anti-virus scan requests and on-access anti-virus scan requests to the virus checkers so that the virus checkers perform on-demand anti-virus scanning concurrent with on-access anti-virus scanning;

which includes grouping the on-demand anti-virus scan requests into chunks, each of the chunks including multiple ones of the on-demand anti-virus scan requests, and for each chunk, distributing the multiple ones of the on-demand anti-virus scan requests over the virus checkers.

9. The method as claimed in claim 8, wherein the on-access anti-virus scan requests are produced in response to user access to files.

10. The method as claimed in claim 8, wherein the on-demand anti-virus scan requests are produced in response to a system administrator requesting a scan of files within a specified file system.

11. The method as claimed in claim 8, which includes inhibiting the distribution of the multiple ones of the on-demand anti-virus scan requests from at least one of the chunks to the virus checkers until completion of anti-virus scanning for the anti-virus scan requests in a prior one of the chunks.

12. A method of operating a plurality of virus checkers for on-demand anti-virus scanning concurrent with on-access anti-virus scanning, the method comprising:

combining on-demand anti-virus scan requests and on-access anti-virus scan requests in a virus scan request queue; and

a pool of threads distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers, each anti-virus scan request on the virus scan request queue being serviced by a respective one of the threads in the pool of threads,

which includes grouping the on-demand anti-virus scan requests into chunks, each of the chunks including multiple ones of the on-demand anti-virus scan requests, and for each chunk, checking whether the number of anti-virus scan requests on the virus checking queue is less than a threshold, and upon finding that the number of anti-virus scan requests on the virus checking queue is less than the threshold, placing said each chunk on the virus scan request queue.

13. The method as claimed in claim 12, wherein the on-access anti-virus scan requests are produced in response to user access to files.

14. The method as claimed in claim 12, wherein the on-demand anti-virus scan requests are produced in response to a system administrator requesting a scan of files within a specified file system.

15. The method as claimed in claim 12, which includes inhibiting the placement of at least one of the chunks onto the virus scan request queue until completion of anti-virus scanning for the anti-virus scan requests in a prior one of the chunks.

16. A virus checking system comprising:

a plurality of virus checkers for on-demand anti-virus scanning concurrent with on-access anti-virus scanning;

a virus scan request queue; and

at least one processor coupled to the virus checkers and the virus scan request queue for sending virus scan requests from the virus scan request queue to the virus checkers, said at least one processor being programmed for placing on-demand anti-virus scan requests and on-access anti-virus scan requests onto the virus scan request queue, and for distributing the on-demand anti-virus scan requests and the on-access virus scan requests from the virus scan request queue to the virus checkers.

17. The virus checking system as claimed in claim 16, wherein said at least one processor and said virus scan request queue are in a file server, and the virus checkers are separate from the file server.

18. The virus checking system as claimed in claim 16, wherein said at least one processor is programmed to place each on-access request onto the virus scan request queue in response to user access of a respective file.

19. The virus checking system as claimed in claim 16, wherein said at least one processor is programmed to produce the on-demand anti-virus scan requests in response to a system administrator requesting a scan of files within a specified file system.

20. The virus checking system as claimed in claim 16, wherein said at least one processor is programmed to execute multiple threads for distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers, each anti-virus scan request on the virus scan request queue being serviced by a respective one of the threads in the pool of threads.

21. The virus checking system as claimed in claim 16, wherein said at least one processor is programmed for giving the on-access anti-virus scan requests priority over the on-demand anti-virus scan requests by inhibiting the placement of on-demand anti-virus scan requests onto the virus scan request queue when the number of anti-virus scan requests on the virus scan request queue reaches a threshold, and not inhibiting the placement of on-access anti-virus scan requests onto the virus scan request queue when the number of anti-virus scan requests on the virus scan request queue reaches the threshold.

22. The virus checking system as claimed in claim 16, wherein said at least one of the processors is programmed for grouping the on-demand anti-virus scan requests into chunks, each of the chunks including multiple ones of the on-demand anti-virus scan requests, and placing the chunks onto the virus scan request queue.

23. The virus checking system as claimed in claim 22, which includes inhibiting the placement of at least one of the chunks onto the virus scan request queue until completion of anti-virus scanning for the anti-virus scan requests in a prior one of the chunks.

24. A virus checking system comprising:

a plurality of virus checkers for on-demand anti-virus scanning concurrent with on-access anti-virus scanning; and

a file server coupled to the virus checkers for sending virus scan requests to the virus checkers, the file server including a virus scan request queue, and the file server being programmed for placing on-demand anti-virus scan requests and on-access anti-virus scan requests onto the virus scan request queue; and for executing multiple threads for distributing the on-demand anti-virus scan requests and the on-access anti-virus scan requests from the virus scan request queue to the virus checkers, each anti-virus scan request on the virus scan request queue being serviced by a respective one of the threads in the pool of threads, the file server further being programmed for grouping the on-demand anti-virus scan requests into chunks, each

of the chunks including multiple ones of the on-demand anti-virus scan requests, and for consecutively placing the chunks onto the virus scan request queue.

25. The virus checking system as claimed in claim 24, wherein the file server is programmed for producing the on-access anti-virus scan requests in response to user access to files.

26. The virus checking system as claimed in claim 24, wherein the file server is programmed to produce the on-demand anti-virus scan requests in response to a system administrator requesting a scan of files within a specified file system.

27. The virus checking system as claimed in claim 24, wherein the file server is programmed for checking for each chunk whether the number of anti-virus scan requests on the virus checking queue is less than a threshold, and upon finding that the number of anti-virus scan requests on the virus checking queue is less than the threshold, placing said each chunk on the virus scan request queue.

28. The virus checking system as claimed in claim 24, wherein the file server is programmed for inhibiting the placement of at least one of the chunks onto the virus scan request queue until completion of anti-virus scanning for the anti-virus scan requests in a prior one of the chunks.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.